

Feature Similarity in using AHC and KNN Algorithm for Text Segmentation based on Text Clustering

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the KNN algorithm and the AHC algorithm which consider the feature similarities, for automating the text segmentation with its combination with the text clustering. In the previous work, even if the KNN version was applied to the text segmentation, a domain should be decided as an input text tag in advance, manually. In this research, a text group is clustered based on its contents, and a text is segmented into subtexts based on their contents, by selecting a classifier which corresponds to its most similar cluster. The AHC algorithm and the KNN algorithm which considers the feature similarities are used respectively for the text clustering and the text segmentation. The goals of this research are to automate the text segmentation and to improve the performance of both tasks, using the two algorithms.

I. INTRODUCTION

The text segmentation is the process of segmenting a text into subtexts based on the topic transition. A boundary is put between paragraphs where the topic is changed, and it is necessary to judge whether the topic in the current paragraph is different from or almost same to its next paragraph. In specifying the text segmentation, an adjacent paragraph pair is set as an entity, and the task is interpreted into the classification of each entity into boundary or continuance. A boundary is put between paragraphs in each pair which is classified into boundary. Subtexts which are outputs of the text segmentation are viewed as independent texts.

The motivation of this research is the necessity of defining domains manually for designing the text segmentation system. The text segmentation is interpreted into a binary classification of paragraph pairs for each domain; the text segmentation system is composed with binary classification systems as many as domains. The process of designing the text segmentation system is to predefine the domains depending on prior knowledge and subjective views, collect sample paragraph pairs domain by domain, and allocate a binary classifier for each domain. Another motivation of this research is the validation of the KNN algorithm and the AHC algorithm which consider the feature similarities as the excellent approaches to the text segmentation and the text clustering. This research is intended to automate the process of predefining domains by connecting the text segmentation with the text clustering.

In this research, we propose that the KNN algorithm and the AHC algorithm which consider the feature similarities to the text segmentation and the text clustering which relate to each other. In this research, we prepare the text collection where each text is segmented into subtexts logically. We define the similarity metric between two numerical vectors which considers the feature similarities and modify the KNN algorithm and the AHC algorithm based on the similarity metric as the approaches to both tasks. The collection is segmented into clusters by the text clustering, sample paragraph pairs which are labeled one of the two categories are collected, and the binary classifier is trained with them for each cluster. A domain is automatically assigned to an input text by its similarities with the cluster prototypes for selecting a corresponding classifier.

Let us mention some benefits from this research. The poor discrimination among sparse vectors is solved by considering the feature similarities in computing the similarity between two numerical vectors. The better performances of the text segmentation and the text clustering are expected by adopting the versions of the KNN algorithm and the AHC algorithm which are modified based on the similarity metric. We avoid defining domains depending on prior knowledge or subjective views by automating the domain predefinition through the text clustering. We are free from deciding manually a domain of an input text by computing its similarities with the cluster prototypes.

Let us mention some remaining tasks as further discussions on this research. Only essential features may be selected in computing the feature similarities for reducing the computation cost. More advanced machine learning algorithms and deep learning algorithms may be modified into the versions which consider the feature similarities. The text segmentation system which relates to the text clustering may be implemented as a real program or a library. The text clustering and the text classification relate to the text segmentation for improving their performances.

II. PROPOSED WORKS

A. Encoding Process

This section is concerned with the process of encoding a text into a numerical vector. Words are used as features for encoding a text so, and some are selected from words in a corpus. The similarity between words is computed based on texts with their collocations and is used as the similarity between features. The similarities among features are considered for computing the semantic similarity between texts. In this section, we describe the process of encoding a text into a numerical vector and the process of computing the similarity between texts.

The process of encoding texts into numerical vectors is illustrated in Figure 1. The entire text collection is indexed into a list of words which are feature candidates. Some words are selected as the features by criteria among the feature candidates in the second step. The weights of the selected features in the text are computed as their relationships with the text, as elements of the numerical vector. Refer to [2] for studying the process and the selection criteria in detail.

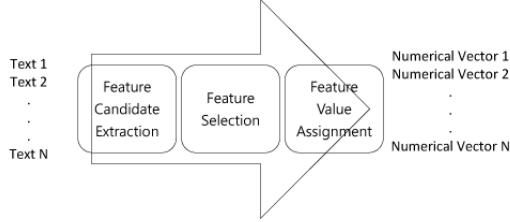


Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_i, f_j)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

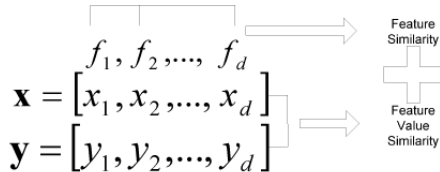


Figure 2. Frame of Computing Similarity between Numerical Vectors

Let us mention the process of computing the similarity between numerical vectors as the similarity between texts. The similarity between the features, f_i and f_j , is notated

by $\text{sim}(f_i, f_j) = s_{ij}$. The similarity between the features is computed by equation (1),

$$s_{ij} = \text{sim}(t_i, t_j) = \frac{2 \times \#(t_i \wedge t_j)}{\#(t_i) + \#(t_j)}, \quad (1)$$

where $\#(t_i \wedge t_j)$, is the number of texts which include both words, t_i and t_j , $\#(t_i)$ is the number of texts which include the word, t_i , and $\#(t_j)$ is the number of texts which include the word, t_j , and the similarity matrix to the d features is constructed as a $d \times d$ square matrix as follows:

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}.$$

The similarity between the two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (2),

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^d \sum_{j=1}^d s_{ij} x_i y_j}{d \|\mathbf{x}\| \|\mathbf{y}\|} \quad (2)$$

Equation (2) is used for computing the similarity between a novice text and a sample text in the KNN algorithm.

Let us make some remarks on the encoding process and the computation of the similarity between two texts. A text is encoded into a numerical vector by the feature candidate extraction, the feature extraction, and the feature value assignment. The similarity between features which are given as words is computed by equation (1). The similarity between numerical vectors which represent texts is computed by equation (2). In future, we consider computing the similarities among some features rather than all features for improving the computation speed.

B. Feature Similarity based KNN Algorithm

This section is concerned with the KNN algorithm which considers the feature similarities. The version was initially proposed as the approach to the text classification by Jo in 2019 [3]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between a novice vector and a training example. The labels of its nearest neighbors are voted with their identical weights for decoding the label of a novice vector. This section is intended to describe the initial version of the KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into numerical vectors, and they are notated by a set $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. A novice word is encoded into a numerical vector notated by $\mathbf{x}$. Its similarities with the numerical vectors which represent the sample words are computed by

equation (2), as $\text{sim}(\mathbf{x}, \mathbf{x}_1), \text{sim}(\mathbf{x}, \mathbf{x}_2), \dots, \text{sim}(\mathbf{x}, \mathbf{x}_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

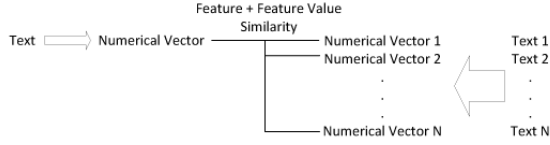


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (3),

$$Nr = \text{Select}_k(Tr, \mathbf{x}), Nr \subseteq Tr \quad (3)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (4),

$$Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}\} \quad (4)$$

The elements in the set, Nr , are used for voting their labels in the next step.

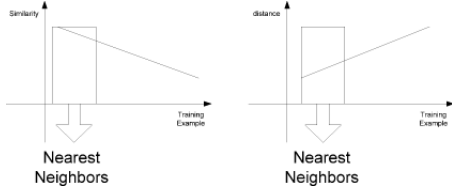


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(\mathbf{x}_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, $\mathbf{x}$, is decided by equation (5),

$$\text{label}(\mathbf{x}) = \arg\max_{j=1}^m \text{Count}(Nr, c_j) \quad (5)$$

The process of deciding the label of a novice item by equation (5), is called voting.

Let us make some remarks on the KNN algorithm which considers the feature similarities. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors.

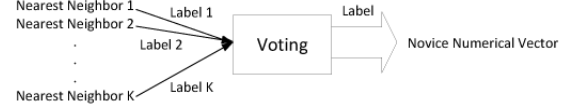


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the text segmentation system which adopts the KNN algorithm which considers the feature similarities. In Section II-B, we described the proposed version of KNN algorithm as the approach to the text segmentation. It is viewed as a binary classification which classifies a paragraph pair into boundary or continuance. This section is intended to describe the text segmentation system with respect to its function and its architecture.

The process of gathering sample paragraph pairs and classifying a novice one, is illustrated in Figure 6. Because even a same paragraph pair may be classified differently depending on the domain, the task is called domain dependent classification. Sample paragraph pairs are gathered domain by domain, and each pair is labeled with boundary or continuance. The paragraph pairs are taken from texts which are tagged with their same domain, each paragraph pair is classified into boundary or continuance. Sample paragraph pairs which are labeled with one of the two categories are encoded into numerical vectors in each domain.

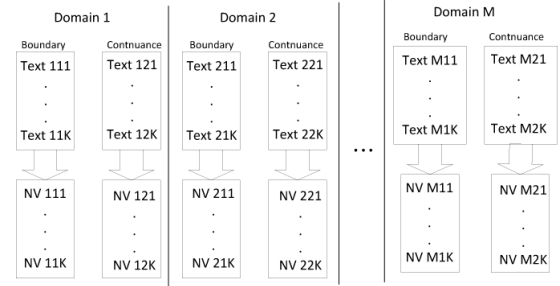


Figure 6. Preparation of Training Examples

The system architecture of the text segmentation system is illustrated in Figure 7. The text partition and sliding module partitions the input text into paragraphs and generate adjacent paragraph pairs by sliding the two sized window, and the encoding module encodes the adjacent paragraph pairs into numerical vectors by the process which is described in Section II-A. The similarity computation module computes the similarities of each paragraph pair with the samples which are labeled with boundary or continuance by the process which is described in Section II-B and

select samples with their highest similarities as the nearest neighbors. The voting module votes the labels of the nearest neighbors for deciding the label of the novice one. In this system, the adjacent paragraphs are classified into boundary or continuance, and a boundary is put between paragraphs in each pair which is classified into boundary.

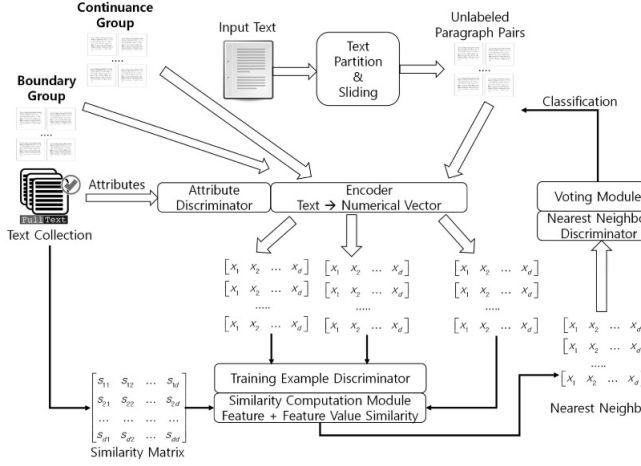


Figure 7. System Architecture

The execution flow of the text segmentation system is illustrated in Figure 8. Texts with same domain are initially collected, adjacent paragraph pairs are extracted from them, and the paragraph pairs are labeled with boundary or continuance. From an input text, adjacent paragraph pairs are extracted, and are encoded into numerical vectors, together with the sample paragraph pairs. The adjacent paragraph pairs are classified into boundary or continuance, and there are two groups of paragraph pairs: boundary group and continuance group. The boundary is marked between paragraphs in each pair in the boundary group, and text is partitioned by the marked boundaries into subtexts as the final output of this system.

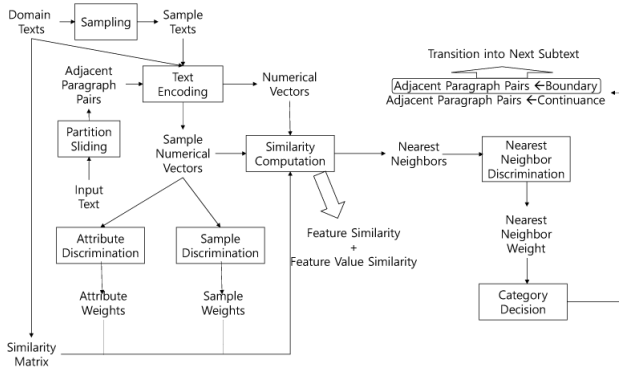


Figure 8. System Flow

Let us make some remarks on the system architecture and the execution flow of the text segmentation system.

In this research, the text segmentation is viewed as the binary classification of adjacent paragraph pairs and the similarity metric between numerical vectors is proposed. The KNN algorithm is modified by the similarity metric, and the version is adopted for implementing the text segmentation system. The system architecture and the execution flow are presented; this indicates that this research stay in the step of making the general design of the system. In next research, we try to make the detail design and the implementation of the entire text segmentation system.

D. Connection with Text Clustering

This section is concerned with the connection of the text segmentation with the text clustering. In the previous section, we mentioned the text segmentation as an instance of domain dependent classification. The domains are automatically constructed from the text collection by clustering texts. The AHC algorithm which is proposed in [1] is used for implementing the text clustering. This section is intended to describe the text segmentation system which is connected with the text clustering for automating the construction of domains.

The architecture of the text clustering system which was proposed in [1] is illustrated in Figure 9. The texts in the group are encoded into numerical vectors, and the AHC algorithm which is proposed in [1] is adopted as the approach. It starts with singletons as many as items, computes the similarities of all possible cluster pairs, and merge the cluster pair with its highest similarity into one cluster. The two steps are iterated until the number of clusters reach the desired number. We may consider replacing the AHC algorithm by its variants for improving the speed and the performance.

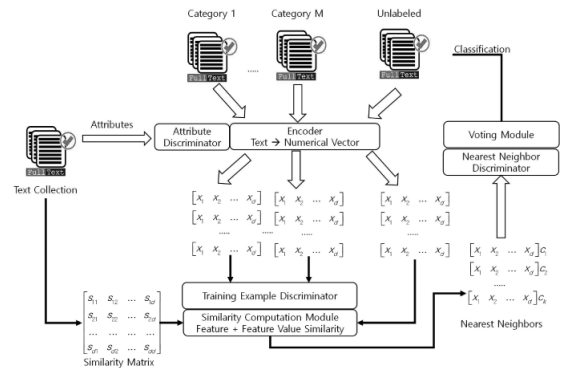


Figure 9. Text Clustering System

The connection of the text segmentation with the text clustering is illustrated in Figure 10. The M domains are defined by clustering texts in a group, initially and automatically. A novice text is given as the input, and its domain, domain D , is decided by its similarities with domains. A classifier which corresponds to domain D is nominated and classify each

adjacent paragraph pair in a novice text into boundary or continuance. The M text segmentation systems are viewed as independent ones, each of which classifies each paragraph pair into one of the two categories.

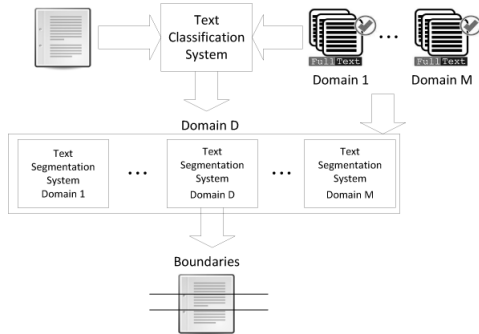


Figure 10. Text Segmentation System connected with Text Clustering

Let us make some remarks on the connection of the text segmentation with the text clustering. It is the process of segmenting a text group into subgroups based on their content-based similarities. The role of text clustering is to define the domains initially in the architecture of connecting the text segmentation with the text clustering. In each domain, sample paragraph pairs are generated, and its corresponding classifier is trained with them. In the future research, we consider using the text segmentation for clustering texts in the opposite direction.

REFERENCES

- [1] T. Jo, "Clustering Texts using Feature Similarity based AHC Algorithm", 5993-6003, Journal of Intelligent and Fuzzy Systems, 5, 2018.
- [2] T. Jo, "Text Mining: Concepts and Big Data Challenge", Springer, 2019.
- [3] T. Jo, "Text Classification using Feature Similarity based K Nearest Neighbor", 13-21, AS Medical Science, 3, 2019.
- [4] T. Jo, "Content based Segmentation of News Articles using Feature Similarity based K Nearest Neighbor", 61-64, The Proceedings of 19st International Conference on Information and Knowledge Engineering, 2019.